Package ‘owmr’

October 14, 2022

Title OpenWeatherMap API Wrapper
Version 0.8.2
Date 2019-12-12
Maintainer Stefan Kuethe <crazycapivara@gmail.com>

Description Accesses OpenWeatherMap's (owm) <https://openweathermap.org/> API. 'owm' itself is a service providing weather data in the past, in the future and now. Furthermore, 'owm' serves weather map layers usable in frameworks like 'leaflet'. In order to access the API, you need to sign up for an API key. There are free and paid plans. Beside functions for fetching weather data from 'owm', 'owmr' supplies tools to tidy up fetched data (for fast and simple access) and to show it on leaflet maps.

URL https://github.com/crazycapivara/owmr/

https://crazycapivara.github.io/owmr/

BugReports https://github.com/crazycapivara/owmr/issues/

Depends R (>= 3.1.2)
Imports magrittr, httr, jsonlite, plyr, tibble, tidyr
License MIT + file LICENSE
Encoding UTF-8
LazyData true

RoxygenNote 6.1.1
Suggests leaflet, whisker, testthat, covr

NeedsCompilation no

Author Stefan Kuethe [aut, cre], Amanda Dobbyn [ctb]

Repository CRAN

Date/Publication 2020-01-11 14:30:02 UTC
add_owm_tiles

Add owm tiles to leaflet map.

Description
Add owm tiles to leaflet map.

Usage
add_owm_tiles(map, layer_name = owm_layers$Temperature_new, ...)

Arguments

map        leaflet map object
layer_name  owm layer name, see owm_layers
...        optional parameters passed to addTiles
add_weather

Value

updated map object

Examples

```
## Not run:
leaflet() %>% add_owm_tiles() %>%
  addMarkers(data = quakes[1:20, ])
```

---

**add_weather**

Add weather data to leaflet map.

Description

Add weather data to leaflet map.

Usage

```
add_weather(map, data, lng = NULL, lat = NULL, icon = NULL,
            template = NULL, popup = NULL, ...)
```

Arguments

- **map** - leaflet map object
- **data** - owm data
- **lng** - numeric vector of longitudes (if NULL it will be taken from data)
- **lat** - numeric vector of latitudes (if NULL it will be taken from data)
- **icon** - vector of owm icon names (usually included in weather column of owm data)
- **template** - template in the form of 
  `<b>{{name}}</b>`
  where variable names in brackets correspond to column names of data (see also render)
- **popup** - vector containing (HTML) content for popups, skipped in case parameter template is given
- **...** - see addMarkers

Value

updated map object
cbind_weather

Flatten weather column in data frame. (DEPRECATED)

Description

Flatten weather column in data frame. (DEPRECATED)

Usage

cbind_weather(data)

Arguments

data data frame containing weather column

Value

data frame with flattened weather (data)

Examples

## Not run:
get_forecast("Kassel") %>% cbind_weather()

## End(Not run)
find_cities_by_bbox  
Find cities by bounding box.

**Description**
Get current weather data for a number of cities within a given bounding box.

**Usage**
`find_cities_by_bbox(bbox = c(12, 32, 15, 37, 10), ...)`

**Arguments**
- `bbox` bounding box, numeric vector of the form (lon-left, lat-bottom, lon-right, lat-top, zoom)
- `...` see [https://openweathermap.org/current](https://openweathermap.org/current)

find_cities_by_geo_point

**Description**
Get current weather data for a number of cities around a given geo point.

**Usage**
`find_cities_by_geo_point(lat, lon, cnt = 3, ...)`

**Arguments**
- `lat` latitude of geo point
- `lon` longitude of geo point
- `cnt` number of cities
- `...` see owm api documentation

**Value**
list

**See Also**
- `find_city`
find_city

Find city by name or coordinates.

Description

Either search for city by name or fetch weather data for a number of cities around geo point.

Usage

find_city(city = NA, ...)

Arguments

- city: city name (and country code)
- ...: see owm api documentation, pass lat and lon to search by coordinates

Value

list of weather data for matches

See Also

- find_cities_by_geo_point

Examples

```r
## Not run:
find_city("London,UK")
find_city(lat = 51.50853, lon = -0.12574, cnt = 5)
```

## End(Not run)
flatten

*Flatten list. (DEPRECATED)*

**Description**

Flatten list. (DEPRECATED)

**Usage**

`flatten(data)`

**Arguments**

- `data`: list returned from owm

**Value**

flattened list

**Examples**

```r
## Not run:
get_current("Rio de Janeiro") %>% flatten() %>%
tidy_up()
## End(Not run)
```

---

`flatten_weather`

*Parse weather column to (single) data frame. (DEPRECATED)*

**Description**

Parse weather column to (single) data frame. (DEPRECATED)

**Usage**

`flatten_weather(x)`

**Arguments**

- `x`: weather column (NOT name)

**Value**

data frame
get_current

Get current weather data for given city.

Description

Get current weather data for given city.

Usage

get_current(city = NA, ...)

Arguments

city  
city name or id

...  
see own api documentation, you can also skip parameter city and pass lat  
(latitude) and lon (longitude) or zip (zip code) instead

Value

list

Examples

## Not run:
get_current("London", units = "metric")
get_current(2643741, lang = "DE")
get_current(lon = -0.09184, lat = 51.51279)
get_current(zip = "94040,US")

## End(Not run)
get_current_for_group

Get current weather data for multiple cities.

Description
Get current weather data for multiple cities.

Usage
get_current_for_group(city_ids, ...)

Arguments
- city_ids: numeric vector containing city ids
- ...: see owm api documentation

Value
list

See Also
- owm_cities dataset in order to lookup city ids

Examples
## Not run:
city_ids = c(2831088, 2847639, 2873291)
result <- get_current_for_group(city_ids)
result$cnt == nrow(result$list)
weather_frame <- result$list
## End(Not run)

get_forecast

Get 3h forecast data.

Description
Get 3h forecast data.

Usage
get_forecast(city = NA, ...)
get_forecast_daily

Arguments

city  city name or id

... see owm api documentation, you can also skip parameter city and pass lat (latitude) and lon (longitude) or zip (zip code) instead

Value

list

Examples

## Not run:
result <- get_forecast("Kassel", units = "metric")
names(result)
get_forecast("London", cnt = 10)
get_forecast(lat = -22.90278, lon = -22.90278, cnt = 3, units = "metric")

## End(Not run)

get_forecast_daily  Get daily forecast data up to 16 days.

Description

Get daily forecast data up to 16 days.

Usage

generate_forecast_daily(city = NA, ...)

Arguments

city  city name or id

... see owm api documentation, you can also skip parameter city and pass lat (latitude) and lon (longitude) or zip (zip code) instead

Value

list

Examples

## Not run:
# 9 day forecast
result <- get_forecast_daily("London", cnt = 9)
forecast_frame <- result$frame

## End(Not run)
Description
Get icon url.

Usage
get_icon_url(icon)

Arguments
icon icon name as returned by owm

Value
icon url

Examples
## Not run:
forecast <- get_forecast("London")$list
weather <- flatten_weather(forecast$weather)
icons <- get_icon_url(weather$icon)

## End(Not run)

owmr
owmr - An R interface to access OpenWeatherMap’s API

Description
In order to access the API, you need to sign up for an API key at https://openweathermap.org/.
For optional parameters (…) in functions see https://openweathermap.org/api/

Examples
## Not run:
# first of all you have to set up your api key
owmr_settings("your_api_key")

# or store it in an environment variable called OWM_API_KEY (recommended)
Sys.setenv(OWM_API_KEY = "your_api_key") # if not set globally

# get current weather data for "Kassel" with temperatures in °C
get_current("Kassel", units = "metric")
owmr_as_tibble

Parse owmr response to tibble.

Description

Parse owmr response to tibble.

Usage

owmr_as_tibble(resp, simplify = TRUE)

## S3 method for class 'owmr_weather'
owmr_as_tibble(resp, simplify = TRUE)

## Default S3 method:
owmr_as_tibble(resp, simplify = TRUE)

## S3 method for class 'owmr_forecast_daily'
owmr_as_tibble(resp, simplify = TRUE)

Arguments

resp response object returned from functions like get_current or get_forecast
simplify return tibble only?

Value

list containing tibble or tibble only (simplify = TRUE)
**owmr_settings**  

Description

Set api key. Internally it calls `Sys.setenv` to store the api key in an environment variable called `OWM_API_KEY`.

Usage

`owmr_settings(api_key)`

Arguments

- `api_key` owm api key

Examples

```r
## Not run:
owmr_settings(api_key = "your-api-key")

## End(Not run)
```

**owm_cities**  

Description

A dataset containing city ids and coordinates to be used in queries.

Usage

`owm_cities`

Format

data frame with 74071 rows and 4 variables:

- `id` city id
- `nm` city name
- `lat` latitude
- `lon` longitude
- `countryCode` two letter country code

Source

[http://bulk.openweathermap.org/sample/city.list.json.gz](http://bulk.openweathermap.org/sample/city.list.json.gz)
owm_layers

List of available owm weather map layers.

Description
List of available owm weather map layers.

Usage
owm_layers

Format
An object of class list of length 16.

See Also
https://openweathermap.org/api/weathermaps

parse_columns

Apply functions to columns.

Description
Apply functions to columns.

Usage
parse_columns(data, functions_)

Arguments
data data frame
functions_ named list where keys correspond to column names

Value
updated data frame

Examples

```r
## Not run:
parse_dt <- function(x){as.POSIXct(x, origin = "1970-01-01")}
forecast <- get_forecast("Kassel")$list
forecast %<>% parse_columns(list(dt = parse_dt))
## End(Not run)
```
**remove_prefix**

Remove prefixes from column names.

**Description**
Remove prefixes from column names.

**Usage**

```r
remove_prefix(data, prefices, sep = ".")
```

**Arguments**
- **data**: data frame
- **prefices**: vector of prefixes to be removed from column names
- **sep**: prefix separator

**Value**
Data frame with updated column names.

**Examples**

```r
x <- data.frame(main.temp = 1:10, sys.msg = "OK", cnt = 10:1)
names(x)
remove_prefix(x, c("main", "sys")) %>% names()
```

---

**search_city_list**

Look up coordinates and city id in owm’s city list.

**Description**
Search owm_cities dataset by city name and country code.

**Usage**

```r
search_city_list(city, country_code = "")
```

**Arguments**
- **city**: city name (regex)
- **country_code**: two letter country code (AU, DE, ...), use `country_code = ""` as wildcard

**Value**
Data frame with matches
See Also

owm_cities dataset

Examples

search_city_list("London", "GB")
search_city_list("London")
search_city_list("Lond")

tidy_up

Tidy up owm data. (DEPRECATED)

Description

Calls tidy_up passing data$list as data argument.

Usage

tidy_up(data, ...)

Arguments

data result returned from owm containing data frame in data$list
...

Value

data with updated data frame (data$list)

See Also

tidy_up_

Examples

## Not run:
get_forecast("London") %>% tidy_up()

## End(Not run)
tidy_up_  

Tidy up own data. (DEPRECATED)

Description

Tidy up own data. (DEPRECATED)

Usage

```r
 tidy_up_(data, flatten_weather_ = TRUE, use_underscore_ = TRUE,
         remove_prefix_ = c("main", "sys"))
```

Arguments

data  
data frame

flatten_weather_  
see `flatten_weather`

use_underscore_  
substitute dots in column names with underscores

remove_prefix_  
prefixes to be removed for shorter column names (remove_prefix_ = NULL will keep all prefixes)

Value

updated data frame

See Also

`tidy_up`,
`remove_prefix`,
`use_underscore`

Examples

```r
## Not run:
result <- find_city("Malaga")
result$list %>% tidy_up_()

# keep dots in column names
result$list %>% tidy_up_(use_underscore_ = FALSE)

# keep all prefixes
result$list %>% tidy_up_(remove_prefix_ = NULL)

## End(Not run)
```
### use_underscore

*Substitute dots in column names with underscores.*

**Description**

Substitute dots in column names with underscores.

**Usage**

```r
code
use_underscore(data)
```

**Arguments**

- **data** data frame

**Value**

data frame with updated column names

**Examples**

```r
code
names(airquality)
use_underscore(airquality) %>% names()
```

### Render operator

*Render operator.*

**Description**

Vectorizes function `whisker.render`.

**Usage**

```r
code
template %$$% data
```

**Arguments**

- **template** template
- **data** data frame where column names correspond to variables names in template

**Value**

rendered template

**NOTE:** Because `whisker` does not support variable names including dots, a *dot* in column names is replaced by an *underscore*. Therefore, you must use an underscore in the template text for variables including dots.
See Also

`whisker.render`

Examples

```r
vars <- data.frame(a = 1:3, b = 23:21)
"a = {{a}} and b = {{b}}" %$$% vars
```
Index

* datasets
  owm_cities, 13
  owm_layers, 14
%$$%. 18

add_owm_tiles, 2
add_weather, 3
addMarkers, 3
addTiles, 2
cbind_weather, 4

find_cities_by_bbox, 5
find_cities_by_geo_point, 5, 6
find_city, 5, 6
flatten, 7
flatten_weather, 7, 17

get_current, 8, 12
get_current_for_group, 9
get_forecast, 9, 12
get_forecast_daily, 10
get_icon_url, 11

leaflet, 3

owm_cities, 9, 13, 15, 16
owm_layers, 2, 14
owmr, 11
owmr_as_tibble, 12
owmr_settings, 13

parse_columns, 14

remove_prefix, 15, 17
render, 3
render (%$$%), 18

search_city_list, 15
Sys.setenv, 13

tidy_up, 16, 17